```
// Computer Program Listing Appendix Under 37 CFR 1.52(e)
// Translator
// Copyright (c) 2004. Borland Software Corporation All Rights Reserved.
class TranslatorManager {
  Hashtable TypeMap = new Hashtable();
  void RegisterTranslator(Translator translator) {
    Type actualType = translator.GetActualType();
    Hashtable subMap = (Hashtable) TypeMap[actualType];
    if(subMap == null) {
      subMap = new Hashtable();
      TypeMap[type] = subMap;
    }
    foreach(string formalType in translator.GetFormalTypes()) {
      subMap[formalType] = translator;
    }
  }
  object LookupInMap(Type actualType, Hashtable map) {
    object result = map[actualType];
    if(result != null) {
      return result;
    }
    // look for a mapping in the interfaces supported by the actualType
    foreach(Type t in actualType.GetInterfaces()) {
      object result = map[t];
      if(result != null) {
        return result;
      }
    }
    // look for a mapping in the base types of the type
    for(Type t = actualType.BaseType; t != null; t = t.BaseType) {
      object result = map[t];
      if(result != null) {
        return result;
      }
    }
    return null;
  }
  Translator LookupTranslator(Type actualType, string formalType) {
    Hashtable subMap = (Hashtable)
      LookupInMap(actualType, TypeMap);
    return (Translator) subMap[formalType];
  }
}
interface Translator {
  /// Returns the C# datatype that is produced or written
  /// by this Translator. This method will be used to determine
  /// which translator to use when an object of a given C# type
  /// is written to the stream.
  Type GetActualType();
  /// Create an instance of the C# datatype supported by
```

```
  /// this Translator
  object Create();
  /// Read in the marshaled data for this datatype from
  /// the input stream.
  void Read(object obj, InputStream input);
  /// Write the state of an instance of this datatype
  void Write(object obj, OutputStream output);
  /// This method returns a list of type IDs that
  /// represent the list of formal types that this
  /// Translator supports.
  string[] GetFormalTypes();
}
/// Provides a translator for constructing, reading and writing
/// instances of type System.Collections.ArrayList from or to a
/// data stream.
class ArrayListTranslator : Translator {
  Type GetActualType() {
    return typeof(System.Collections.ArrayList);
  }
  object Create() {
    return new System.Collections.ArrayList();
  }
  void Read(object obj, InputStream input){} // details omitted
  void Write(object obj, OutputStream output){} // details omitted
  string[] GetFormalTypes() {
    return new string[] {
      "java.util.ArrayList",
      "java.util.AbstractList",
      "java.util.AbstractCollection",
      "java.util.Collection",
      "java.util.List",
      "java.util.RandomAccess",
      "java.lang.Cloneable",
    };
  }
}
```